Table of Contents

Troubleshooting

Common Login Failures or Issues

DRAFT

This article is being reviewed for completeness and technical accuracy.

- **SSH Known-Hosts Error**

  SSH offers the ability to verify the identity of the remote host to which you are connecting. A successful host verification indicates that your SSH client has estabilished a secure connection with the SSH server and no intermediate machines have access to that connection.

  The identity of the remote host can be verified by checking the host public key of the remote host stored either in the system-wide */etc/ssh/ssh_known_hosts* file (or */etc/ssh_known_hosts* for some systems) or your personal *~/.ssh/known_hosts* file on your localhost.

  SSH has three ways it can react to an unrecognized or changed SSH host key, based on the value of the *StrictHostKeyChecking* variable in either the system-wide */etc/ssh/ssh_config* file (or */etc/ssh_config* for some systems) or your personal *~/.ssh/config* file :

  - `StrictHostKeyChecking=no`

    This is the most insecure setting as it will blindly connect to the server. It will add the server's key if it's not present locally, and if the key has changed it will add the key without asking.
  - `StrictHostKeyChecking=ask`

    With this setting, if you have no host key for the server, it will show you the fingerprint and ask you to confirm. If you connect and the key does not match, it will prevent you from logging in, and will tell you where to find the conflicting key inside the *known_hosts* file.
  - `StrictHostKeyChecking=yes`

    This is the most secure setting. If you have no host key for this server, then it will prevent you from logging in at all.

  If *StrictHostKeyChecking* is set to *yes* and you get errors similar to the following the first time you log in to one of the SFEs, the simple solution is to add **-o**

"stricthostkeychecking=ask" in your ssh command. Sfe1 is used in this example.

Sample Error:

```
your_localhost% ssh sfe1.nas.nasa.gov

WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
11:9f:ae:09:56:2d:45:66:8e:9a:df:15:52:d6:88:5e.
Please contact your system administrator.
Add correct host key in /Users/userid/.ssh/known_hosts2 to get rid
of this message.
Offending key in /etc/ssh_known_hosts2:24
RSA host key for sfe1.nas.nasa.gov has changed and you have
requested strict checking.
No RSA host key is known for sfe1.nas.nasa.gov and you have
requested strict checking.
Host key verification failed.
```

Solution:

```
your_localhost% ssh -o "stricthostkeychecking=ask" sfe1.nas.nasa.gov

-----------------------------------------------------------------
The authenticity of host 'sfe1.nas.nasa.gov (198.9.4.3)'
can't be established.
RSA key fingerprint is
11:9f:ae:09:56:2d:45:66:8e:9a:df:15:52:d6:88:5e.
Are you sure you want to continue connecting (yes/no)? yes
-----------------------------------------------------------------
Warning: Permanently added 'sfe1.nas.nasa.gov,198.9.4.3' (RSA) to
the list of known hosts.

-----------------------------------------------------------------
Plugin authentication
Enter PASSCODE: type your passcode
```

- **Common SecurID Passcode Problems**
    - ♦ If you cannot log in after you created your new PIN,

```
your_localhost% ssh sfe1
user@sfe1's password:
Authenticated with partial success.
Plugin authentication
Enter PASSCODE:
Plugin authentication
You may create your own PIN or accept a server assigned PIN.
Would you like to create your own new PIN (yes/no)? yes
Plugin authentication
Enter your new PIN (exactly 8 alphanumeric characters)
```

```
New PIN:
Confirm new PIN:
Plugin authentication
Enter PASSCODE:
Plugin authentication
Enter PASSCODE:
Permission denied, please try again.
user@sfe1's password:
Permission denied, please try again.
user@sfe1's password:
Permission denied ().
```

it is possibly that you did not successfully complete the "New-Pin Process". You might have used a special character in your PIN. Your PIN MUST consist of **EXACTLY 8 alphanumeric characters**, with at least one letter and at least one number, but no special characters.

♦ Each SecurID 6-digit tokencode can be used only once. If you try to use a tokencode that has just been used, you will be prompted again to enter a new passcode.

```
your_localhost% ssh sfe1.nas.nasa.gov
PAM Authentication
Enter PASSCODE: enter PIN and a tokencode which was just used
PAM Authentication
Enter PASSCODE: ener PIN and a new token code
```

♦ If you failed to provide a correct PIN + tokencode in a few consecutive attempts, your SecurID fob will be temporarily disabled. NAS Help Desk can help you unlock your fob (even for fobs provided by your local center). Call NAS Help Desk at 1-800-331-USER (8737) or 1-650-604-4444 for assistance.

- **Common Passthrough Problems**

If you set up SSH passthrough correctly, you should be prompted for your SecurID passcode only, and be transferred to the desired host. The following describes a few common problems and their solutions:

♦ If the authentication failed when you entered the passcode, it is possible that you have different usernames between your localhost and the NAS systems and you did not include your NAS username in your *.ssh/config* file.

There are two ways to include your NAS username in the *.ssh/config* file:

◊ If you use the *.ssh/config* file for connecting to multiple computer sites, then you should add your NAS username to the ProxyCommand lines for corresponding NAS hosts. For example:

```
Host lou1
ProxyCommand ssh username@sfe1.nas.nasa.gov /usr/local/bin/ssh-proxy
```

In this case, you will still need to issue your ssh command as the following example in order to avoid being prompted for a password:

```
%ssh nas_username@lou1
```

◊ If you use the *.ssh/config* file only for connecting to NAS, then you can simply add the following at the beginning of your *.ssh/config* file:

```
User nas_username
```

You do no need to add your NAS username to the ProxyCommand line. In addition, you can simply use the following command and won't be prompted for a password:

```
%ssh lou1
```

♦ If you are prompted for password in addition to the passcode, there are multiple possible causes:

◊ As described above, you may have a different username and need to use

```
your_localhost% %ssh nas_username@lou1
```

◊ Either your home directory or the *.ssh/authorized_keys* file under your NAS account have write permission for group or others. You need to correct the permission so that they have write permission only to you.

◊ Either one or both of the following files is missing:

· *.ssh2/authorization* of sfe[1,2]
· *.ssh/authorized_keys* of the NAS HECC host that you want to connect to

♦ If you are prompted for passphrase in addition to the passcode, likely, you did not use the commands "ssh-agent" and "ssh-add ~/.ssh/id_rsa" to forward your private key before you issue the ssh command.

- **Incorrect Ciphers**

A cipher is an algorithm for performing encryption or decryption. The SSH client and server must have a matching cipher in order to successfuly verify the keys. If you get an error similar to the following:

```
no matching cipher found: client blowfish-cbc
server aes128-cbc,aes192-cbc,aes256-cbc,3des-cbc
```

check your *./ssh/config* file or */etc/ssh/ssh_config* file on your localhost and add the appropriate ciphers.

```
To modify /etc/ssh_config, system administrator's privilege
may be needed.
```

- **Password Expiration**

  Your NAS password is valid for 90 days. An email is sent to you by NAS reminding you to change your password. If it has expired, you can still log in. However, you will be prompted to change it right away.

  ```
  This policy will change from 90 days to 60 days in the near
  future.
  ```

- **Account Expiration or Deactivation**

  Your NAS account is active if you have a valid project and a valid account request form on file at NAS. If your account has expired, it will be removed from NAS database and the */etc/passwd* files. When this happens, no login will be allowed.

  Please note that the account request form has to be filled out once every year. When the form has expired, you should receive an email from the NAS account administrator asking you to fill out a new one.

  If you violate NAS security rules such as those listed in the Acceptable Use Statement, your account can be deactived.

Post Login Issues

DRAFT

This article is being reviewed for completeness and technical accuracy.

- **Idle SSH connections are broken after 10-15 minutes**

  If you find that idle SSH connections tend to be dropped after 10 to 15 minutes and your are using a router or other device that implements Network Address Translation (NAT), such as a DSL router, you may want to enable the *ServerAliveInterval* parameter in your *~/.ssh/config*.

  The issue is that some devices that implement NAT will drop information from their state tables if a TCP connection has been idle for just a few minutes. In such cases, it may be necessary to ensure that the connection is never truly idle for an extended period.

  The *ServerAliveInterval* parameter (supported in OpenSSH 3.8 and later) causes the SSH client to periodically send an encrypted query to determine if the server is still responsive. The packets will make the connection appear to be active and can prevent the NAT state table entry for the connection from timing out.

  For example, adding the following to your *~/.ssh/config* will cause an encrypted query to be sent every five minutes:

  ```
  ServerAliveInterval 5m
  ```

  ```
  NOTE: You may need to experiment with the length of the
  interval to determine an appropriate value to prevent your
  connection from being broken. If you use the NAS config
  template, ServerAliveInterval is set to 10 minutes.
  ```

- **SSH connection is periodically broken even when connection is active**

  During an SSH session, the client and server will re-negotiate the keys used to encrypt the session every so often. Very old versions of OpenSSH did not implement this feature, which can cause a problem when connected to the SFEs. Your connection may appear to hang or may abort with an error message such as the following:

  ```
  Dispatch protocol type 20
  ```

  If you encounter this issue, you will need to upgrade to a newer version of OpenSSH.

```
NOTE: Other implementations that were derived from the OpenSSH
distribution, such as SUN Secure Shell, are likewise known to
exhibit this problem.
```

- **Slow performance transferring data**

  While OpenSSH performs reasonably for local data transfers, the performance will tend to be reduced due to the latency of long-haul network connections.

  Depending on the severity of the impact you may have several options to improve this situation.

  1. Upgrade to OpenSSH 4.7 or later

     If you are using a version of OpenSSH that is older than 4.7, you may see an improvement in file transfer performance by upgrading to OpenSSH 4.7 or later. This is due to the use of a larger channel buffer introduced in that version.

  2. Use an HPN-enabled version of OpenSSH

     The High-Performance Networking (HPN) patch set maintained by the Pittsburgh Supercomputing Center (PSC) allows the channel buffer used by OpenSSH to grow as needed. This may be useful in cases where OpenSSH 4.7 does not yield satisfactory performance.
  3. Enabling compression

     In cases where you are seeing very poor performance (under 1.2 MB/s) and the data you are transferring will compress well, you may wish to enable compression.

     You can do this by adding -C to your scp or sftp command-line.

  Read Tips for File Transfers for more recommendations.

Common Reasons for Being Unable to Submit Jobs

DRAFT

This article is being reviewed for completeness and technical accuracy.

There are several common reasons why you might not be able to successfully submit a job to PBS:

- Resource request exceeds resource limits

  qsub: Job exceeds queue resource limits

  Reduce your resource request to below the limit or use a different queue.
- AUID or GID not authorized to use a specific queue

  If you get the following message after submitting a PBS job:

  qsub: Unauthorized Request

  it is possible that you tried submitting to a queue which is accessible only to certain groups or users. You can check the "qstat -fQ" output and see if there is an acl_groups or a acl_users list. If your group or username is not in the lists, you will have to submit to a different queue."

  If your GID has no allocation left, you will also get this error. See the section 'Not enough or no allocation left' below for more information.

- AUID not authorized to use a specific GID

  If you get the following message after submitting a PBS job:

  qsub: Bad GID for job execution

  it is possible that your AUID has not been added to use allocations under a specific GID. Please consult with the principal investigator of that GID and ask him/her to submit a request to support@nas.nasa.gov to add your AUID under that GID.
- Queue is disabled

  If you get the following message after submitting a PBS job

  qsub: Queue is not enabled

  you should submit to a different queue which is enabled.
- Not enough or no allocation left

An automated script is used to check if a GID is over its allocation everyday. If it does, that GID is removed from PBS access control list and users of that GID will not be able to submit jobs.

Users can check the amount of allocations remaining using the acct_ytd command. In addition, if you see in your PBS output file some message regarding your GID allocation usage is near its limit or is already over, ask your PI to request for more allocation.

Once the request for more allocation is approved and added to your account, an automatic hourly script will add your GID back to the PBS access control list.

Common Reasons Why Jobs Won't Start

Once you've successfully submitted your job, there may be several common reasons why it might not run:

- **The job is waiting for resources**

  Your job may be waiting for resources, due to one of the following:
    - All resources are tied up with running jobs, and no other jobs can be started.
    - PBS may have enough resources to run your job, however, there is another higher priority job that needs more resources than what is available, and PBS is draining the system (including not running any new jobs) in order to accommodate the high-priority job.
    - Some users submit too many jobs at once (e.g., more than 100), and the PBS scheduler becomes swamped with sorting jobs and is not able to start jobs effectively.
    - In the case when you request your job to run on a specific node or group of nodes, your job is likely to wait in the queue longer than if you do not request specific nodes.
- **Your mission share has run out**

  Your mission shares have been used up. The available resources that you saw belong to other missions, which can be borrowed. However, your job may have requested a wall-time that is too long (more than 4 hours for Pleiades), which prevents your job from borrowing the resources.

  See also, Mission Shares Policy on Pleiades.
- **The system is going into dedicated time**

  When dedicated time (DED) is scheduled for hardware and/or software work, the PBS scheduler will not start a job if the projected end time runs past the beginning of the DED time. If you are able to reduce the requested wall-time so that your job will finish running prior to DED time, then your job can then be considered for running. To change the wall-time request for your job, follow the example below :

  ```
  %qalter -l walltime=hh:mm:ss jobid
  ```
- **Scheduling is turned off**

  Sometimes job scheduling is turned off by control room staff or a system administrator. This is usually done when there are system or PBS issues that need to be resolved before jobs can be scheduled to run. When this happens, you should see the following message near the beginning of the "qstat -au your_userid" output.

  ```
  +++Scheduling turned off.
  ```
- **Your job has been placed in "H" mode**

A job can be placed on hold either by the job owner or by someone who has root privilege, such as a system administrator. If your job has been placed on hold by a system administrator, you should get an email explaining the reason for the hold.

- **Your home filesystem or default /nobackup filesystem is down**

  When a PBS job starts, the PBS prologue checks to determine whether your home filesystem and default /nobackup are available before executing the commands in your script. If your default /nobackup filesystem is down, PBS can not run your job and it will put the job back in the queue. If your PBS job does not need any file in that filesystem, you can tell PBS that your job will not use the default /nobackup so that your job can start running. For example, if your default is /nobackupp10 (for Pleiades), you can add the following in your PBS script:

  ```
  #PBS -l /nobackupp10=0
  ```

Using pdsh_gdb for Debugging Pleiades PBS Jobs

DRAFT

This article is being reviewed for completeness and technical accuracy.

A script called *pdsh_gdb*, created by NAS staff Steve Heistand, is available on Pleiades under */u/scicon/tools/bin* for debugging PBS jobs **while the job is running**.

Launching this script from a Pleiades front-end node allows one to connect to each compute node of a PBS job and create a stack trace of each process. The aggregated stack trace from each process will be written to a user specified directory (by default, it is written to ~/tmp).

Here is an example of how to use this script:

```
pfe1% mkdir tmp
pfe1% /u/scicon/tools/bin/pdsh_gdb -j jobid -d tmp -s -u nas_username
```

More usage information can be found by launching pdsh_gdb without any option:

```
pfe1% /u/scicon/tools/bin/pdsh_gdb
```